EARTHQUAKE ENGINEERING AND STRUCTURAL DYNAMICS, VOL. 25, 117–128 (1996)

MODELLING UNKNOWN STRUCTURAL SYSTEMS THROUGH THE USE OF NEURAL NETWORKS

A. G. CHASSIAKOS

College of Engineering, California State University, Long Beach, CA 90840, U.S.A.

AND

S. F. MASRI

School of Engineering, University of Southern California, Los Angeles, CA 90089, U.S.A.

SUMMARY

This paper explores the potential of using neural networks to identify the internal forces of typical systems encountered in the field of earthquake engineering and structural dynamics. After formulating the identification task as a neural network learning procedure, the method is applied to a representative chain-like system under deterministic and stochastic excitations. The neural network based identification method provides very good results for general classes of multi-degree-of-freedom structural systems. The range of validity of the approach is demonstrated, and some application issues are discussed for (a) partially known multi-degree-of-freedom systems and (b) completely unknown systems.

KEYWORDS: identification; modelling; neural; nonparametric; dynamics; systems

1. INTRODUCTION

System identification refers to any systematic way of deriving or improving models for dynamic systems through the use of experimental data. It is an area of considerable importance in structural engineering which has been gaining increasing attention over the last decade or so. Some representative publications on the subject are available in the work of Beck,[1] Ibanez,[2] Masri and Caughey,[3] Natke,[4] Masri and Werner[5] and the IMAC Proceedings.[6] The methods of system identification provide a means of utilizing laboratory and field testing to improve dynamic modelling capabilities for civil infrastructure systems such as high-rise buildings, bridges and dams.

For example, by systematically utilizing dynamic test data from a structure, rather than relying on theory alone, models can be derived which provide more accurate response predictions for dynamic loads on the structure which are produced by wind or earthquakes. Another application is to continually update the model through vibration monitoring of the structure to provide a convenient method for defect identification or damage assessment.[7–9]

The potential for using active control approaches to reduce the response of large civil structures under arbitrary dynamic environments, such as earthquakes, has drawn a considerable amount of interest worldwide. Among the key research topics in this area is the development of system identification approaches that can cope with the challenging nature of physical structures encountered in the structural mechanics and earthquake engineering fields.[10] Since the model structure in many practical dynamics problems is by no means clear, an increasing amount of attention is being devoted to non-parametric identification methods. These methods do not identify the physical parameters of the system (such as mass, stiffness, etc.) but instead identify the parameters of a mathematical model which fits the input/output data.

The present paper introduces a new non-parametric identification method for unknown dynamic systems undergoing arbitrary earthquake-type excitation. The method is based on the use of artificial neural

CCC 0098–8847/96/020117–12

Received 23 June 1993
Revised 15 August 1995

networks as system identifiers. Artificial neural networks are the ideal choice in cases when real time processing of large amounts of data is required because of their inherent massive parallelism, fault tolerance and learning capabilities.

In some very recent publications neural nets are used for the detection of structural damage,[11] and for the identification of single-degree-of-freedom structural systems with linear or non-linear restoring force characteristics.[12-15] In Chassiakos and Masri,[16] a multi-degree-of-freedom system has been identified and subsequently validated under stochastic earthquake-like excitation. This is the type of system studied in the present paper. Moreover, other issues presented and discussed in this paper are: the network size and topology, the network training algorithms, validation of the identified model and its prediction capabilities.

2. NEURAL NETWORK FORMULATION

2.1. Neural network approach

An Artificial Neural Network (ANN) is a system with inputs and outputs, composed of a number of similar non-linear processing elements. These processing elements operate in parallel and are arranged in patterns similar to the patterns found in biological neural nets. The processing elements or nodes are connected to each other by adjustable weights.[17,18] Changing these weights will change the input/output behaviour of the network, hence the following is a natural goal for such a system: choose the weights of the net in such a way as to achieve a desired input/output relationship. To achieve this goal, systematic ways of adjusting the weights have to be developed, which are referred to as training or learning algorithms. A neural net is characterized by the type of processing elements (nodes), the network topology and the learning algorithm.

A typical node sums n weighted inputs $u_1, u_2, \ldots, u_n$ and a bias term b and passes the result through a non-linear function $\gamma(\cdot)$ as shown in Figure 1:

$$v = \gamma(\bar{v}); \quad \bar{v} = \sum_{i=1}^{n} w_i u_i + b$$

The choice of the non-linearity depends on the particular application for which the network is being used. In cases when it is required that the non-linearity is differentiable, the hyperbolic tangent function (sigmoid)

$$\gamma(x) = \frac{1 - e^{-\alpha x}}{1 + e^{-\alpha x}}; \quad \alpha > 0$$

is commonly used.

The network topology depends on the way in which the nodes are connected to each other and on the input and output vectors. According to their topology, neural networks can be classified as

(a) single-layer networks, when only one layer of nodes is present, the output layer, or
(b) multi-layer networks, when the nodes are arranged in more than one layer.

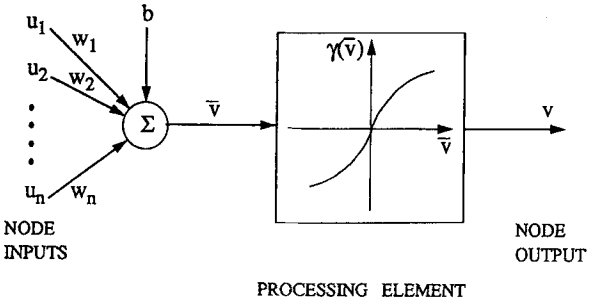

Figure 1. A processing element

Moreover, a network can be characterized as:

(1) A feedforward network, if there is no feedback to previous layers from the output of the subsequent layers.
(2) Recurrent network, if such a feedback connection exists.

In Figure 2 a three-layer feedforward neural network is shown, similar to topology to the ones that will be used in this paper. The various inputs and outputs in each layer are related by the following equations:

$$\text{Layer 1:} \quad \bar{v} = W^1 u; \quad v_i = \gamma(\bar{v}_i), \; i = 1, \ldots, p$$

$$\text{Layer 2:} \quad \bar{z} = W^2 v; \quad z_i = \gamma(\bar{z}_i), \; i = 1, \ldots, q$$

$$\text{Layer 3:} \quad \bar{y} = W^3 z; \quad y_i = \gamma(\bar{y}_i), \; i = 1, \ldots, m$$

It is seen that the output vector y is a non-linear function of the input vector u and of the weights $\{w_{ij}\}$. It is precisely this non-linear dependence that makes the neural net a powerful tool for approximating arbitrary functions, hence a potential tool for systems identification.

2.2. The training algorithms

The term training or learning algorithm refers to a systematic procedure for adjusting the weights in the network in order to achieve a desired input/output relationship. In the case of 'supervised' learning the network is being presented pairs of input vectors and desired output vectors (u^r, y_d^r), where the superscript r ranges over all pairs used to train the network and the subscript d stands for 'desired' output vector. During training the network learns to associate the input vector u^r with the output vector y_d^r. For a given set of weights, if the network is presented an input u^r it will produce an output y^r, which should be identical or very close to y_d^r if the training was successful.

A general feature of 'supervised' learning algorithms is that a performance criterion $E = f(y, y_d)$ is evaluated, where y is the actual network output vector and y_d is the desired output vector. Then the weights w_{ij} of the net are adjusted in such a way as to reduce the value of this error criterion. The various learning algorithms reported in the literature differ in the ways this adjustment is being done.

As an example, the back propagation training algorithm uses a square error defined as

$$E = \frac{1}{2} \sum_r \sum_{k=1}^{m} (y_k^r - y_{kd}^r)^2$$

where the index r ranges over all training patterns.

The performance criterion is a measure of the distance between the output vectors y^r and the desired outputs y_d^r. The criterion E is a function of the actual output values y_k^r, which in turn depend on the values of the network weights $\{w_{ij}\}$ and the input patterns. Because of this dependence of y_k^r on $\{w_{ij}\}$, the criterion E can be reduced if the weights w_{ij} are adjusted appropriately. In order to find a set of parameters (weights)

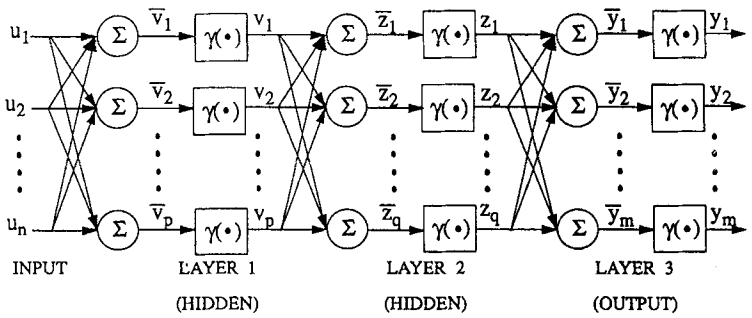

Figure 2. A feedforward three-layer network

that will reduce E, we need to calculate the gradient of E with respect to the parameters $\{w_{ij}\}$ and then adjust the parameter set in the direction of the negative gradient. The back propagation algorithm calculates this gradient, i.e. the partial derivatives $\partial E/\partial w_{ij}$ of the criterion E with respect to the elements w_{ij} of the weight matrices. The algorithm starts at the output layer and 'propagates' the results backwards to the first layer. After the gradient matrices have been computed, the weights are adjusted in the negative gradient direction, so that the error criterion E is reduced. A typical weight w_{ij} (which could belong to any layer) is adjusted from its old value w_{ij}^{old} to its new value w_{ij}^{new} according to

$$w_{ij}^{\text{new}} = w_{ij}^{\text{old}} - \eta \frac{\partial E}{\partial w_{ij}^{\text{old}}}$$

The stepsize η is termed the 'learning rate'. This η is usually chosen as constant during training, but it could also be adjustable if this facilitates the training process.

Detailed descriptions of the back propagation algorithm can be found in Narendra and Parthasarathy,[19] where a graphical representation of the algorithm is given, and in McLelland and Rumelhart[20] for an analytical formulation.

3. APPLICATIONS

3.1. System description

A methodology based on a neural network formulation is developed and applied to identify a structural chain-like system undergoing deterministic as well as random excitation.

Consider the N-degree-of-freedom chain-like system (Figure 3) that is governed by the following equation of motion:

$$M\ddot{y}(t) + g(y(t), \dot{y}(t)) = -m\ddot{s}(t) \tag{1}$$

where $\ddot{s}(t)$ is the base acceleration vector, y is the vector of relative displacements with respect to the base, $g(y, \dot{y})$ is the vector of restoring forces, $M = \text{diag}\{m_1, m_2, \ldots, m_N\}$ is the mass matrix and $m = [m_1, m_2, \ldots, m_N]^{\text{T}}$.

This formulation represents general classes of structural systems, such as systems with linear force/deflection characteristics, non-linear systems having polynomial-form non-linearities, etc. Assume that the experimental measurements for $\ddot{s}(t)$ and $\ddot{y}(t)$ are available and that the corresponding relative displacement $y(t)$ and velocity $\dot{y}(t)$ vectors can be found by direct measurements or through integration of $\ddot{y}(t)$. If the measurements are taken at discrete times t_k, then the following notation is used:

$$y_k = y(t_k); \quad \dot{y}_k = \dot{y}(t_k); \quad \ddot{y}_k = \ddot{y}(t_k); \quad \ddot{s}_k = \ddot{s}(t_k)$$

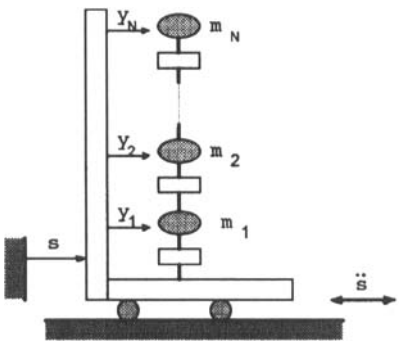

Figure 3. Base-excited three-degree-of-freedom chain system used for identification studies

In the development that follows we study two modelling and identification problems:

(i) The system is partially known: the vector of restoring forces $g(y, \dot{y})$ is identified from input/output data, assuming that the mass vector $m = [m_1, m_2, \ldots, m_N]^T$ is known.
(ii) The system is completely unknown (i.e. not even the masses are known: the system behaviour is modelled on the basis of input/output data only).

3.2. Partially known system

In this case it is assumed that the masses m_i, $i = 1, \ldots, N$, are known, or can be accurately estimated, but the restoring force vector function $g(y, \dot{y})$ is unknown. The neural network is trained to identify a model of $g(y, \dot{y})$. The system identification is done on the basis of input/output data, where the inputs are the measured values of $y_k, \dot{y}_k$ and the outputs are the calculated values of $g_k = (-M\ddot{y}_k - m\ddot{s}_k)$. The block diagram of the network identifier is shown in Figure 4(a). During the training phase, the neural network is presented repeatedly with the sequence of input vectors $\{[y_k, \dot{y}_k]^T\}$ and the sequence $\{g_k\}$ of desired output vectors. The training algorithm adjusts the weights of the network in such a way as to reduce the error between the desired output g_k and the actual output of the network $\hat{g}_k$.

After the training procedure is completed, the network is validated as the identifier of system (1): given an *arbitrary* input vector $[y_\alpha, \dot{y}_\alpha]^T$, at time $t = \alpha$, the net should produce an output $\hat{g}_\alpha$ which is very close to the true value of the restoring force vector $g_\alpha = -M\ddot{y}_\alpha - m\ddot{s}_\alpha$.

To test the validity of this procedure, simulations were performed on a three-degree-of-freedom system with linear force–deflection characteristics, under stochastic as well as deterministic excitation. The three-degree-of-freedom model has the following characteristics:

$$\text{Masses:} \quad m_1 = 12; \; m_2 = 12; \; m_3 = 12 \quad (\text{lb s}^2/\text{in})$$

$$\text{Stiffness:} \quad k_1 = 7214; \; k_2 = 14\,900; \; k_3 = 14\,900 \quad (\text{lb/in})$$

$$\text{Damping:} \quad c_1 = 21; \; c_2 = 60; \; c_3 = 60 \quad (\text{lb s/in})^\dagger$$

A three-layer feedforward neural network is trained to identify the unknown system and then its modelling capabilities are validated. Training is done by the back propagation algorithm. The network has six input nodes (corresponding to the three components of the displacement vector y and to the three components of the velocity vector $\dot{y}$), three output nodes (corresponding to the three components of the restoring force vector $g(y, \dot{y})$) and four and four nodes in the first and second hidden layers, respectively.

During the training phase, system (1) is excited by a swept-sine excitation, given by $\ddot{s}(t) = \sin[\omega_0 + (\omega_F - \omega_0)(t/t_F)]t$, where $\omega_0 = 10$ rad/s, $\omega_F = 15$ rad/s and $t_F = 10{\cdot}0$ s. The base acceleration $\ddot{s}(t)$ as well as the base velocity and displacement are shown in Figure 5. In this and subsequent plots, consistent units of in, in/s, in/s^2 and matching force units are used (1in = 25·4 mm). The simulation time step is $\Delta t = 0{\cdot}010$ s,

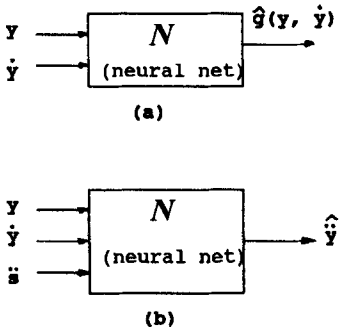

Figure 4. Block diagram of neural identifier

† 1 lb s^2/in = 175·1 kg; 1 lb/in = 175· 1N/m; 1lb s/in = 175·1 N s/m

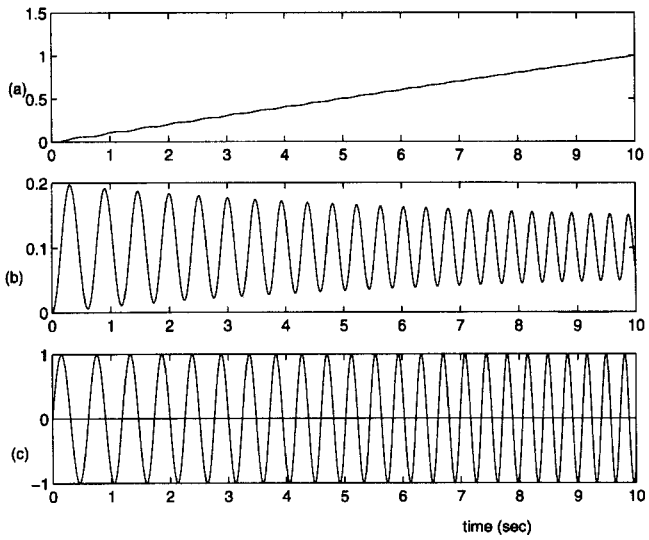

Figure 5. Swept-sine base excitation used for identification: (a) displacement; (b) velocity; (c) acceleration

which provides us with 1001 data points. However, not all of the 1001 data points are used to train the network. Only 10 per cent of the data points (i.e. 100 points) are used for training. These training points are obtained by sampling the structural response $\ddot{y}_k$ and $\dot{y}_k, y_k$, at time t_k, with $\Delta t_k = 0{\cdot}10$ s. This sampling provides us with the input sequence to the neural net $\{[y_k, \dot{y}_k]^T\}$ as well as the output sequence $\{g_k\} = \{-M\ddot{y}_k - m\ddot{s}_k\}$.

The initial values of the network weights are chosen randomly. Subsequently the weights are adjusted according to the back propagation algorithm, as described in Section 2, until an acceptable error is obtained. At this point the values of the weights are frozen, and the training phase is over: the network has 'learned' the system. Figures 6(a)–6(c) show a comparison of the neural network output $\hat{g}$ and the actual system output g, after training is completed. The two curves shown in each of Figures 6(a)–6(c) are virtually indistinguishable. It is seen that the network is performing extremely well in matching the system's response.

At this point the need for scaling of the network input/output data should be mentioned: each network node contains a hyperbolic tangent function as the node non-linearity, which saturates at the values $+1{\cdot}0$ or $-1{\cdot}0$ if the input to the node has large enough magnitude. In order to avoid simultaneous saturation of all the nodes in the network, the inputs and outputs should be scaled down. This scaling does not change the internal network processing nor does it alter the training algorithm. In the simulation at hand, the values of the output sequence $\{g_k\}$ have been divided by a factor of 200, whereas the input sequence values $\{y_k, \dot{y}_k\}$ have not been scaled down at all.

During the next phase, the neural network is validated as a system identifier. System (1) is excited by a random excitation. The random base acceleration as well as the base velocity and displacement are shown in Figure 7. The measurements of y_k and $\dot{y}_k$ are used as the input sequence to the network. It is noted that this random input *has not been used during the training phase*; hence this is the first time that the network 'sees' such an input. The weights of the net are the ones already obtained from the training phase. The outputs $\hat{g}$ of the network are shown in Figures 8(a)–8(c). Again the two curves shown in each of Figures 8(a)–8(c) are virtually indistinguishable. A comparison of $\hat{g}$ to the actual vector of restoring forces g shows that the neural net performs extremely well even when given random inputs, on which it has never been trained.

3.3. Completely unknown system

In this case the mass vector $m = [m_1, m_2, \ldots, m_N]^T$ as well as the vector of restoring forces $g(y, \dot{y})$ are unknown. A neural network will be used to model the system behaviour on the basis of input/output data only. The data $\ddot{s}_k$, $\dot{y}_k$ and y_k are used as the input sequence to the network, whereas the relative base acceleration $\ddot{y}_k$ will be the output sequence. The block diagram of Figure 4(b) shows the modelling of the

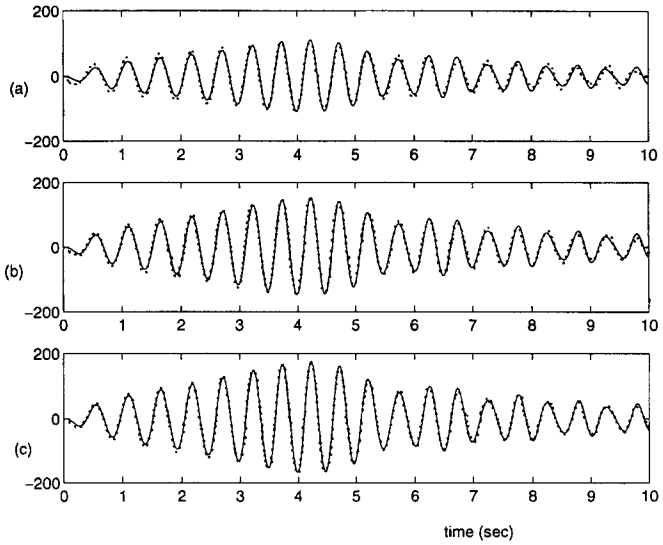

Figure 6. Identification results corresponding to partially known system. Comparison between exact and identified system output under swept-sine base excitation. (—) actual system $g(y, \dot{y})$; (– – –) neural net $\hat{g}(y, \dot{y})$. (a) g_1; (b) g_2; (c) g_3

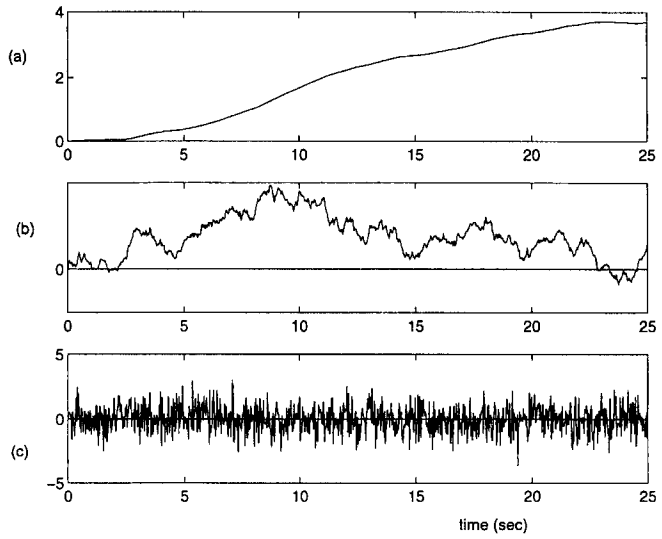

Figure 7. Random base excitation used to validate identification results: (a) displacement; (b) velocity; (c) acceleration

unknown system (1) by a neural network. This modelling represents the system when equation (1) is written in the form

$$\ddot{y}(t) = M^{-1}\{-g(y, \dot{y}) - m\ddot{s}(t)\} \tag{2}$$

where the system parameters m_i, $i = 1, \ldots, N$, and the vector function $g(y, \dot{y})$ are unknown.

Again, during the training phase the neural network is presented repeatedly with the sequence of input vectors $\{[y_k, \dot{y}_k, \ddot{s}_k]^T\}$ and the sequence $\{\ddot{y}_k\}$ of desired output vectors and the back propagation algorithm adjusts the weights of the network so that the error is reduced.

The three-degree-of-freedom system used in Section 3.1 (Figure 3) is used for the simulations in the case of unknown masses as well. The swept-sine and random base excitations are the same as in Section 3.1 (Figures

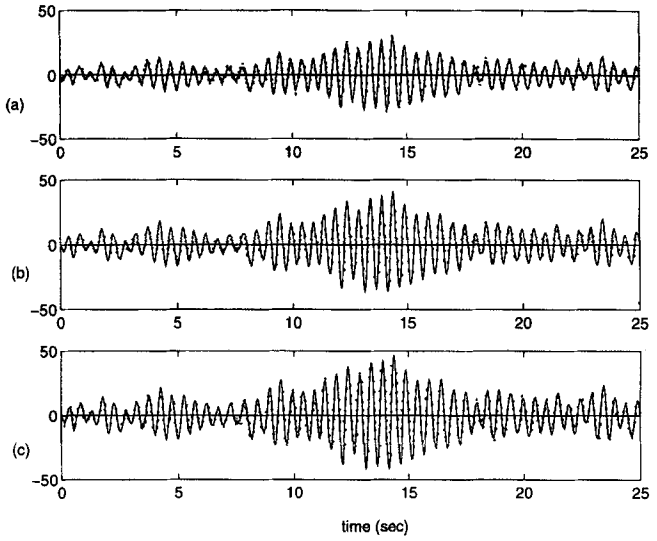

Figure 8. Validation of identification results corresponding to partially known system. Comparison between exact and identified system output under random base excitation.(—) actual system $g(y, \dot{y})$; (- - -) neural net $\hat{g}(y, \dot{y})$. (a) g_1; (b) g_2; (c) g_3

5 and 7). The network used here is a feedforward net with seven input nodes (corresponding to the three components of the displacement vector y, the three components of the velocity vector $\dot{y}$ and the scalar $\ddot{s}$), three output nodes (corresponding to the three components of the acceleration vector $\ddot{y}$), and four nodes in the first and second hidden layers, respectively. The following scaling is performed: $\{y_k, \dot{y}_k, \ddot{s}_k/5\} \leftarrow \{y_k, \dot{y}_k, \ddot{s}_k\}$ and $\{\ddot{y}_k/20\} \leftarrow \{\ddot{y}_k\}$. The network is trained on a combination of swept-sine and random excitation data in a manner similar to that of Section 3.2.

In Figures 9(a)–9(c) the output $\hat{\ddot{y}}$ of the trained network is shown when the system is excited by a swept-sine, and a comparison to the actual relative base acceleration $\ddot{y}$ is done. Figures 10(a)–10(c) show the network output $\hat{\ddot{y}}$ and the actual relative base acceleration $\ddot{y}$ in response to a random base excitation. It is seen that in both cases the network performs very well in modelling the unknown system.

It is noted that the approach of this section does not assume anything about the physical configuration of the structural system (e.g. whether it is chain-like, etc.) or about other system characteristics (e.g. whether it has linear or non-linear force/deflection characteristics). Despite the complete lack of knowledge about the system's parameters, the neural network is able to identify the multi-degree-of-freedom system very well, and this is due to its ability to approximate continuous non-linear functions arbitrarily close, given enough neurons and at least one hidden layer. Moreover, the massive parallelism of multilayer feedforward neural networks enables them to perform the identification procedure very fast, possibly in real time.

3.4. Discussion

As shown in the previous section, the neural network performs very well in modelling a partially known or completely unknown structural system based on input/output information. A question that naturally arises is how many nodes should be chosen, how many layers should be used and what is the effect of these choices on the modelling accuracy of the network. The choice of network size and topology is typically made based on experience and by trial and error. No specific guidelines exist that will enable one to choose a certain network over another without training and testing them first. Some general suggestions can be made, such as we always need at least one hidden layer (other than the input and output layers) in order to obtain a network that is a universal function approximator. Moreover, the decisions about the network choice depend heavily on the particular problem at hand. For the examples presented in this paper, we are dealing with a linear structural system with linear input/output characteristics. Identification of this linear relation through other methods would require about 10–20 parameters, so the neural network used must have at least as many

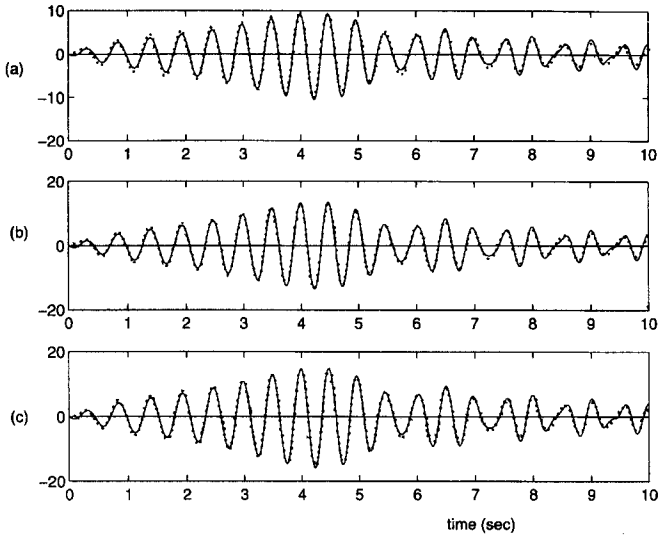

Figure 9. Identification results corresponding to completely unknown system. Comparison between exact and identified system output under swept-sine base excitation. (—) actual system acceleration $\ddot{y}$; (– – –) neural net acceleration $\hat{\ddot{y}}$. (a) $\ddot{y}_1$; (b) $\ddot{y}_2$; (c) $\ddot{y}_3$

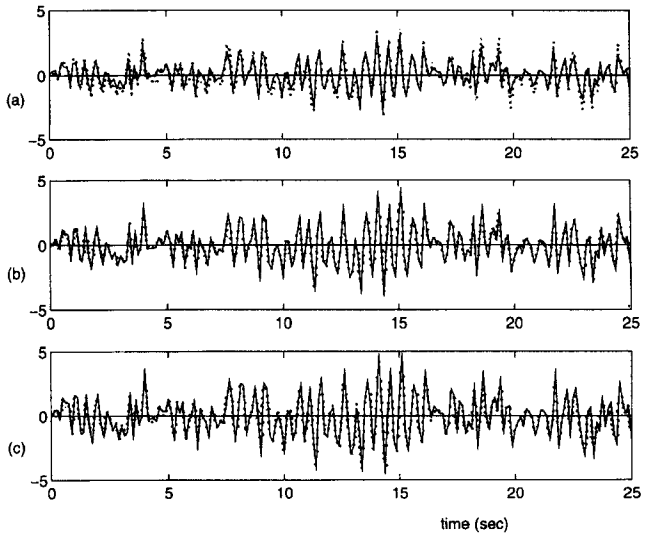

Figure 10. Validation of identification results corresponding to completely unknown system. Comparison between exact and identified system output under random base excitation. (—) actual system acceleration $\ddot{y}$; (– – –) neural net acceleration $\hat{\ddot{y}}$. (a) $\ddot{y}_1$; (b) $\ddot{y}_2$; (c) $\ddot{y}_3$

adjustable weights, and this in turn determines the minimum network size. It is noted, however, that a neural network does not model a system with the minimal number of parameters. On the contrary, the fact that a neural network uses more than the minimal number of parameters provides neural networks with an inherent fault tolerance. Even if a network is partially destroyed (e.g. by severing some connections between network nodes) this will not destory the approximating properties of the network. There may be a degradation in performance, but still the network will perform satisfactorily, provided of course that the number of destroyed connections is relatively small. It should also be mentioned that a network larger in size does not necessarily imply better accuracy on the system identification. As a matter of fact, the authors obtained worse

identification results with a much larger network than with the one presented here. In addition a larger network takes longer time to train.

A graph of the sum of squared errors (criterion E) vs. the number of iterations is shown in Figures 11(a)–11(b), for the case described in Section 3.2. The behaviour of E is typical of the back propagation algorithm. It takes very few iterations to go from a very high initial value ($E \approx 125$) to a much smaller value ($E \approx 30$). Then there is a plateau phase, during which E decreases a little, up to about iteration 130 (Figure 11(a)). Afterwards the error decreases from a value $E = 0{\cdot}056$ at iteration 300 to $E = 0{\cdot}032$ at iteration 1000 (Figure 11(b)). In Figures 12(a)–12(b) density plots of the weight matrices and bias vectors are shown for the partially known system (Figure 12(a)) and the unknown system (Figure 12(b)). The weights within each graph are ordered in a matrix form, e.g. the upper left corner of graph $W1$ represents the element $W^1(1,1)$ of weight matrix W^1. The gray tone of each element is proportional to its numerical value. The vertical bars on the right show the correspondence of the gray scale to the numerical values. The numerical range of the gray scale is the interval $[-1{\cdot}5, 1{\cdot}5]$. As seen in these plots matrix W^1 in part (a) is a 4×6 matrix, and matrix W^1 in part (b) is a 4×7 matrix since this corresponds to the unknown system case. A qualitative comparison of these two matrices shows that W^1 in (a) and the 4×6 submatrix of W^1 in (b) (containing the first six columns of W^1) are qualitatively similar. This is explained by the fact that the inputs to W^1 in (a) and the first six inputs to W^1 in (b) are the same (displacement and velocity information). It is also noted that the bias vectors are qualitatively similar in both (a) and (b) whereas the W^2 and W^3 matrices do not exhibit strong similarities.

As mentioned in Section 3, the data points used for training were only a small fraction of all the available data points (only 10 per cent of the available data). The decision on which sets will be used for training is again based on experience and on the particular problem at hand. For example, in the case of the unknown

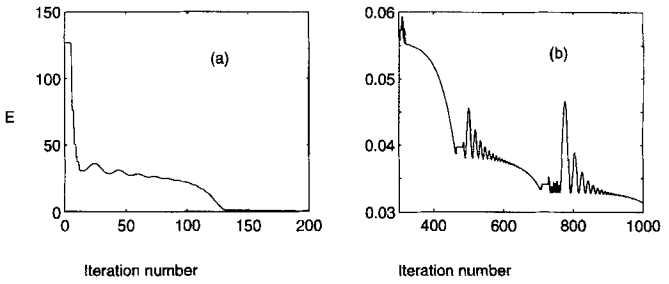

Figure 11. Network training: sum of squared errors (criterion E) vs. number of iterations, for the partially known system; (a) iterations 1–200, (b) iterations 300–1000

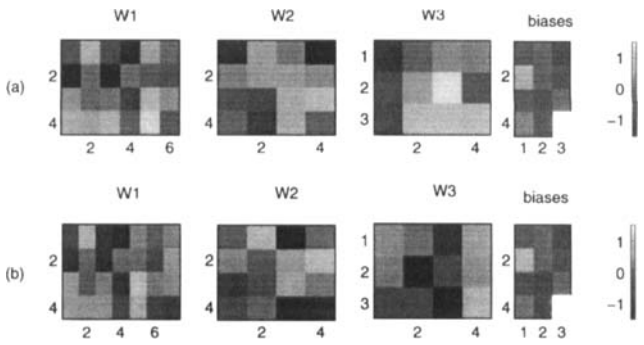

Figure 12. Density plots of the network weight matrices and bias vectors after training: (a) partially known system; (b) completely unknown system

system, the input vectors to the network $[y_k, \dot{y}_k, \ddot{s}_k]$ belong to a seven-dimensional input space, and they must adequately populate the neighbourhood of this space over which the approximation is desired. It is noted that the neural network will perform a good *interpolation* between points in the training set, but it has very poor *extrapolation* properties. It is also noted that choosing the training inputs which are optimal for a particular problem and a particular network is still an open question. Insight into this can be obtained from experience and from the remarks made above.

4. CONCLUSIONS

This exploratory study demonstrates the potential of using neural networks for the identification of the internal (restoring) forces of structural systems undergoing deterministic or random-like excitations. It is shown that employing a three-layer feedforward net is adequate to characterize the internal forces in a linear three-degree-of-freedom chain-like system. The neural net is validated as a system identifier and is able to predict the system's response to random excitations for both cases studied here: (a) system is partially known and (b) system is completely unknown. Plans for future research are to apply the methods of this paper to large structural systems with experimental input/output data. The approach presented here is directly applicable to systems with a high number of degrees of freedom without any modifications. As the dimensionality of the problem increases, so does the size of the approximating neural network. Of course, any *a priori* information about the system can be used to decrease the network size: if e.g. the system can be broken into smaller subsystems, then it is preferable to train smaller networks to approximate each subsystem separately.

ACKNOWLEDGEMENTS

This study was supported in part by a grant from the National Science Foundation, by NASA through the IRA program and by the Carpenters Contractors Cooperation Committee, Inc. The assistance of M. Masri is appreciated. The authors would like to thank the anonymous reviewers for their valuable comments and suggestions.

REFERENCES

1. J. L. Beck, 'Determining models of structures from earthquake records', Earthquake Engineering Research Laboratory, California Institute of Technology, Pasadena, 1978.
2. P. Ibanez, 'Review of analytical and experimental techniques for improving structural dynamic models', *Welding Research Council Bulletin* No. 249, June 1979.
3. S. F. Masri and T. K. Caughey, 'A nonparametric identification technique for nonlinear dynamic problems', *J. appl. mech. ASME* **46**, 433–447 (1979).
4. H. G. Natke (Ed.), *Identification of Vibrating Structures*, Springer, Berlin, 1982.
5. S. F. Masri and S. D. Werner, 'An evaluation of a class of practical optimization techniques for structural dynamics applications', *Earthquake eng. struct. dyn.* **13**, 635–649 (1985).
6. *Proc. modal analysis conf.*, Union College, Schenectedy, New York, 1992.
7. H. G. Natke and J. T. P. Yao (Eds.), *Proc. workshop on structural safety evaluation based on system identification approaches*, Vieweg & Sons, Wiesbaden, 1988.
8. J. Chen (Ed.), *Proc. USAF/NASA workshop on model determination for large space systems*, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, March 1988.
9. J. Garba, (Ed.), *Proc. USAF/NASA workshop on system identification and health monitoring*, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, March 1990.
10. G.W. Housner and S. F. Masri, (Eds.), *Proc. U.S. national workshop on structural control research*, University of Southern California, *Publication CE-9013*, 1990.
11. J. N. Kudva, N. Munir and P. Tan, 'Damage detection in smart structures using neural networks and finite element analysis', *Proc. ADPA/AIAA/ASME/SPIE Conf. on active materials and adaptive structures*, Institute of Physics, IOP Publications, 1992.
12. S. F. Masri, A. G. Chassiakos and T. K. Caughey, 'Structure-unknown non-linear dynamic systems: identification through neural networks', *Smart materials structures* **1**, 45–56 (1992).
13. S. F. Masri, A. G. Chassiakos and T. K. Caughey, 'Identification of nonlinear dynamic systems using neural networks', *J. appl. mech. ASME* **60**, 123–133 (1993).
14. A. G. Chassiakos and S. F. Masri, 'Identification of the internal forces of structural systems using feedforward multilayer networks', *Comput. systems eng.* **2**, 125–134 (1991).

15. A. Chassiakos and S. Masri, 'Neural network based identification of structural systems', *Proc. parallel and distributed computing in engineering systems*, Tzafestas, Borne, Grandinetti, North-Holland, Amsterdam, 1992.
16. A. G. Chassiakos and S. F. Masri, 'Modeling unknown structural systems through the use of neural network', *Proc. 10th world conf. earthquake eng.*, Barcelona, Spain, August 1992.
17. R. P. Lippmann, 'An introduction to computing with neural nets', *IEEE ASSP mag.*, April, 4–22 (1987).
18. D. H. Nguyen and B. Widrow, 'Neural networks for self learning control systems', *IEEE control systems mag.* **10** (3), 18–23 (1990).
19. K. S Narendra and K. Parthasarathy, 'Identification and control of dynamical systems using neural networks', *IEEE trans. neural networks* **1**, 4–27 (1990).
20. J. L. McLelland and D. E. Rumelhart, *Parallel Distributed Processing*, Vol. 1, MIT Press, Cambridge, MA, 1986.